Application Programmer's Guide

Linux Driver for TPRO/TSAT-PCI

Edition 1.1

KSI

a division of DSP*Con*, Inc.

For technical support, contact John Bodino at 888-377-2668

Linux Driver APPLICATION PROGRAMMERS GUIDE
Edition 1.1 —August 12, 2003

A Publication of
KSI

REVISION HISTORY

Revision Date	Revision No.	Revised Section(s)	Comments/Notes
December 20, 2001	1	——	First Edition
August 12, 2003	1.1		Modified the device name to support multiple devices. Documented the wait routines as implemented in the next version

Table of Contents

Chapter One

Overview

This guide provides comprehensive information on the Linux Driver for the KSI TPRO/TSAT-PCI board.

Introduction

The Driver for the KSI TPRO/SAT-PCI Board provides functionality to the board's interfaces and devices.

Product Description

The TPRO/TSAT-PCI performs timing and synchronization functions referenced to an input timecode signal. The board synchronizes its on-board clock to the incoming timecode. The on-board clock's time is also provided as an IRIG-B output. The board includes a time-tag TTL input, a programmable "heartbeat" pulse or squarewave output (with interrupt capability), and a programmable "match" start/stop time output (with interrupt capability)

The TPRO/TSAT-PCI continues to increment time ("freewheel") in the absence of an input timecode. Thus, the board can be used as an IRIG-B timecode generator by setting the initial time via the PCI bus.

The input timecode format (IRIG-B, IRIG-A, or NASA36) is detected automatically. Synchronization to the input timecode is also automatic and can be enabled/disabled via the PCI bus. A propagation delay offset may be specified to compensate for cable delays.

The timecode input is an amplitude-modulated sine wave. An automatic gain control (AGC) circuit permits a wide range of input amplitudes. The timecode input is differential; the board does not reference this signal to ground. A single-ended input (referenced to ground) is also acceptable.

The board can be ordered with option "-M" to synchronize to a one-pulse-per-second (1PPS) input instead of an incoming timecode. In this case, the initial time is programmed via the PCI bus, and the board begins counting on the next 1 PPS pulse.

This page intentionally left blank.

Chapter Two

Installation and Application

Installing the Driver

- Log in to the system as the "root" user
- Type ***"/sbin/insmod tpropci.o"*** at the shell prompt
- Find the driver major number of the tpropci device by typing ***"more /proc/devices"***
- Make a node to the device ***"mknod /dev/tpropciX c <major number> 0"***

  Where X is the board number in the system (0 if the only board in system)

Uninstalling the Driver

- Type ***"/sbin/rmmod tpropci"*** at the shell prompt

Application Example

```
#include <stdlib.h>
#include <stdio.h>

#include <tpro.h>

int main (void)
{
  TPRO_BoardObj *pBoard;
  TPRO_TimeObj tproTime;

  unsigned char rv;

  /**
  ***  open board
  **/

  if (rv = TPRO_open (&pBoard, "/dev/tpropci0"), rv != TPRO_SUCCESS) {
    printf ("Could Not Open Board!! [%d]\n", rv);
    return (1);
  }


  /**
  ***  get time
  **/

  if (rv = TPRO_getTime (pBoard, &tproTime), rv != TPRO_SUCCESS) {
    printf ("Could not retrieve time!! [%d]\n", rv);
    return (1);
  }
  printf ("  Days: %d\n", tproTime.days);
  printf ("  Hours: %d\n", tproTime.hours);
  printf ("  Minutes: %d\n", tproTime.minutes);
  printf ("  Seconds: %f\n", tproTime.secsDouble);
```

```
  /**
  ***  close board
  **/

  if (rv = TPRO_close (pBoard), rv != TPRO_SUCCESS) {
    printf ("Could Not Close Board!! [%d]\n", rv);
    return (1);
  }

  return (0);
}
```

Chapter Three

The TPRO/TSAT-PCI Linux Driver

Header File

```
/*****************************************************************************
  DSPCon TPRO/tSAT - Interface Header

              DSPCon, Inc.
              380 FootHill Road
              Bridgewater, NJ 08807

              e-mail: info@dspcon.com

  (C) Copyright 2001 DSPCon, Inc.  All rights reserved.
  Use of copyright notice is precautionary and does not imply publication
*****************************************************************************/

/*****************************************************************************
  TPRO.H
*****************************************************************************/

#ifndef _defined_TPRO_
#define _defined_TPRO_


/*****************************************************************************
                          SUPPORT CONSTANTS
*****************************************************************************/


/**
***  Heartbeat constants
**/

#define SIG_PULSE      (0xE5)         /*-- heartbeat is a pulse --------*/
#define SIG_SQUARE     (0xE7)         /*-- heartbeat is a squarewave ---*/

#define SIG_NO_JAM     (0)            /*-- start next cycle ------------*/
#define SIG_JAM        (1)            /*-- start immediately -----------*/


/**
***  Match constants
**/

#define MATCH_TIME_START      (0)            /*-- start time ----------*/
#define MATCH_TIME_STOP       (1)            /*-- stop time -----------*/


/**
***  Oscillator frequencies - for Compact PCI Card Only
**/

#define OSC_OUT_OFF           (0)
#define OSC_OUT_1KHZ          (1)
#define OSC_OUT_1MHZ          (2)
#define OSC_OUT_5MHZ          (3)
#define OSC_OUT_10MHZ         (4)



/*****************************************************************************
                               OBJECTS
*****************************************************************************/


/*===========================================================================
                         TPRO BOARD OBJECT
===========================================================================*/
```

```
typedef struct TPRO_BoardObj
{ /*-------------------------------------------------------------------------*/
  int file_descriptor;
  unsigned short devid;

  unsigned short options;

  unsigned char firmware[5];
  unsigned char FPGA[5];
  unsigned char driver[7];
} /*-------------------------------------------------------------------------*/
TPRO_BoardObj;


/*=========================================================================
                        TPRO ALTITUDE OBJECT
=========================================================================*/

typedef struct TPRO_AltObj
{ /*-------------------------------------------------------------------------*/
  float         meters;                     /*-- meters --------------*/
} /*-------------------------------------------------------------------------*/
TPRO_AltObj;


/*=========================================================================
                         TPRO DATE OBJECT
=========================================================================*/

typedef struct TPRO_DateObj
{ /*-------------------------------------------------------------------------*/
  unsigned short year;                      /*-- year ----------------*/
  unsigned char  month;                     /*-- month ---------------*/
  unsigned char  day;                       /*-- day -----------------*/
} /*-------------------------------------------------------------------------*/
TPRO_DateObj;


/*=========================================================================
                 TPRO LONGITUDE/LATTITUDE OBJECT
=========================================================================*/

typedef struct TPRO_LongLat
{ /*-------------------------------------------------------------------------*/
  unsigned short degrees;                   /*-- degrees -------------*/
  float         minutes;                    /*-- minutes -------------*/
} /*-------------------------------------------------------------------------*/
TPRO_LongObj, TPRO_LatObj;


/*=========================================================================
                        TPRO MATCH OBJECT
=========================================================================*/

typedef struct TPRO_MatchObj
{ /*-------------------------------------------------------------------------*/
  unsigned char   matchType;                /*-- start/stop time -----*/

  double         seconds;                   /*-- seconds -------------*/
  unsigned char   minutes;                  /*-- minutes -------------*/
  unsigned char   hours;                    /*-- hours ---------------*/
  unsigned short  days;                     /*-- days ----------------*/
} /*-------------------------------------------------------------------------*/
TPRO_MatchObj;


/*=========================================================================
                        TPRO SATINFO OBJECT
=========================================================================*/

typedef struct TPRO_SatObj
```

```
{ /*-------------------------------------------------------------------------*/
  unsigned char satsTracked;                /*-- num sats tracked ----*/
  unsigned char satsView;                   /*-- num sats in view ----*/
} /*-------------------------------------------------------------------------*/
TPRO_SatObj;


/*=========================================================================
                      TPRO HEARTBEAT OBJECT
=========================================================================*/

typedef struct TPRO_HeartObj
{ /*-------------------------------------------------------------------------*/
  unsigned char signalType;                 /*-- square or pulse -----*/
  unsigned char outputType;                 /*-- jamming option ------*/

  double  frequency;                        /*-- heartbeat freq ------*/
} /*-------------------------------------------------------------------------*/
TPRO_HeartObj;


/*=========================================================================
                        TPRO TIME OBJECT
=========================================================================*/

typedef struct TPRO_TimeObj
{ /*-------------------------------------------------------------------------*/
  double         secsDouble;                /*-- seconds floating pt -*/
  unsigned char  seconds;                   /*-- seconds whole num ---*/
  unsigned char  minutes;                   /*-- minutes -------------*/
  unsigned char  hours;                     /*-- hours ---------------*/
  unsigned short days;                      /*-- days ----------------*/

  unsigned short year;                      /*-- year for CPCI board -*/
} /*-------------------------------------------------------------------------*/
TPRO_TimeObj;


/*=========================================================================
                        TPRO WAIT OBJECT
=========================================================================*/

typedef struct TPRO_WaitObj
{ /*-------------------------------------------------------------------------*/
  int jiffies;                              /*-- # jiffies to wait ---*/

  double  seconds;                          /*-- seconds -------------*/
  unsigned char  minutes;                   /*-- minutes -------------*/
  unsigned char  hours;                     /*-- hours ---------------*/
  unsigned short days;                      /*-- days ----------------*/
} /*-------------------------------------------------------------------------*/
TPRO_WaitObj;

/*=========================================================================
                  TPRO MEM OBJECT FOR PEEK/POKE
=========================================================================*/

typedef struct TPRO_MemObj
{ /*-------------------------------------------------------------------------*/
  unsigned short offset;
  unsigned short value;

  unsigned long l_value;
} /*-------------------------------------------------------------------------*/
TPRO_MemObj;


/***************************************************************************
                           ERROR CODES
***************************************************************************/

#define TPRO_SUCCESS                (0)     /*-- success -------------*/
```

```
#define TPRO_HANDLE_ERR               (1)     /*-- error bad handle ----*/
#define TPRO_OBJECT_ERR               (2)     /*-- error creating obj --*/
#define TPRO_CLOSE_HANDLE_ERR         (3)     /*-- err closing device --*/
#define TPRO_DEVICE_NOT_OPEN_ERR      (4)     /*-- device not opened ---*/
#define TPRO_INVALID_BOARD_TYPE_ERR   (5)     /*-- invalid device ------*/
#define TPRO_FREQ_ERR                 (6)     /*-- invalid frequency ---*/
#define TPRO_YEAR_PARM_ERR            (7)     /*-- invalid year --------*/
#define TPRO_DAY_PARM_ERR             (8)     /*-- invalid day ---------*/
#define TPRO_HOUR_PARM_ERR            (9)     /*-- invalid hour --------*/
#define TPRO_MIN_PARM_ERR            (10)     /*-- invalid minutes -----*/
#define TPRO_SEC_PARM_ERR            (11)     /*-- invalid seconds -----*/
#define TPRO_DELAY_PARM_ERR          (12)     /*-- invalid delay -------*/
#define TPRO_TIMEOUT_ERR             (13)     /*-- device timed out ----*/
#define TPRO_COMM_ERR                (14)     /*-- communication error -*/


/*************************************************************************
                      PUBLIC ROUTINE PROTOTYPES
*************************************************************************/

unsigned char TPRO_open             (TPRO_BoardObj **hnd, char *deviceName);
unsigned char TPRO_close            (TPRO_BoardObj *hnd);

unsigned char TPRO_getAltitude      (TPRO_BoardObj *hnd, TPRO_AltObj *Altp);
unsigned char TPRO_getDate          (TPRO_BoardObj *hnd, TPRO_DateObj *Datep);
unsigned char TPRO_getLatitude      (TPRO_BoardObj *hnd, TPRO_LatObj *Latp);
unsigned char TPRO_getLongitude     (TPRO_BoardObj *hnd, TPRO_LongObj *Longp);
unsigned char TPRO_getSatInfo       (TPRO_BoardObj *hnd, TPRO_SatObj *Satp);
unsigned char TPRO_getTime          (TPRO_BoardObj *hnd, TPRO_TimeObj *Timep);
unsigned char TPRO_resetFirmware    (TPRO_BoardObj *hnd);
unsigned char TPRO_setHeartbeat     (TPRO_BoardObj *hnd, TPRO_HeartObj *Heartp);
unsigned char TPRO_setMatchTime     (TPRO_BoardObj *hnd, TPRO_MatchObj *Matchp);
unsigned char TPRO_setOscillator    (TPRO_BoardObj *hnd, unsigned char *freq);
unsigned char TPRO_setPropDelayCorr(TPRO_BoardObj *hnd, int *us);
unsigned char TPRO_setTime          (TPRO_BoardObj *hnd, TPRO_TimeObj *Timep);
unsigned char TPRO_setYear          (TPRO_BoardObj *hnd, unsigned short *yr);
unsigned char TPRO_simEvent         (TPRO_BoardObj *hnd);
unsigned char TPRO_synchControl     (TPRO_BoardObj *hnd, unsigned char *enbp);
unsigned char TPRO_synchStatus      (TPRO_BoardObj *hnd, unsigned char *status);
unsigned char TPRO_waitEvent        (TPRO_BoardObj *hnd, TPRO_WaitObj *waitp);
unsigned char TPRO_waitHeartbeat    (TPRO_BoardObj *hnd, int *jiffies);
unsigned char TPRO_waitMatch        (TPRO_BoardObj *hnd, int *jiffies);

#endif  // _defined_TPRO_
```

Support Routine Descriptions

TPRO_open

```
unsigned char TPRO_open  (TPRO_BoardObj **hnd, char *deviceName);
```

This routine allocates a TPRO_BoardObj obect, sets a handle to the tpro/tsat, and sets the driver firmware, fpga revision (if applicable), and the driver revision strings.

```
Arguments:  Pointer to TPRO_BoardObj handle
            Device name - "TPROpciX"

Returns:    TPRO_OBJECT_ERR - error allocating board object
            TPRO_HANDLE_ERR - error retrieving handle to device
            TPRO_COMM_ERR   - error communicating with driver
            TPRO_SUCCESS    - success
```

TPRO_close

```
unsigned char TPRO_close (TPRO_BoardObj *hnd);
```

This routine frees the allocated board object and closes the handle to the tpro/tsat device.

```
Arguments:  Pointer to TPRO_BoardObj

Returns:    TPRO_CLOSE_HANDLE_ERR - error closing handle to device
            TPRO_DEVICE_NOT_OPEN  - device is not open
            TPRO_SUCCESS          - success
```

TPRO_getAltitude

```
unsigned char TPRO_getAltitude (TPRO_BoardObj *hnd, TPRO_AltObj *Altp);
```

This routine retrieves the altitude information from the tSAT board. Altitude distance is in meters.

```
Arguments:  Pointer to TPRO_BoardObj
            Pointer to TPRO_AltObj

Returns:    TPRO_INVALID_BOARD_TYPE_ERR - invalid board type for function
            TPRO_COMM_ERR               - error communicating with driver
            TPRO_SUCCESS                - success
```

TPRO_getDate

unsigned char TPRO_getDate (TPRO_BoardObj *hnd, TPRO_DateObj *Datep);

This routine retrieves the current date from the TPRO/tSAT board. The date is in Gregorian Format.

```
Arguments:  Pointer to TPRO_BoardObj
            Pointer to TPRO_DateObj

Returns:    TPRO_INVALID_BOARD_TYPE_ERR - invalid board type for function
            TPRO_COMM_ERR               - error communicating with driver
            TPRO_SUCCESS                - success
```

TPRO_getLatitude

unsigned char TPRO_getLattitude(TPRO_BoardObj *hnd, TPRO_LatObj *Latp);

This routine retrieves the latitude information from the tSAT device.

```
Arguments:  Pointer to TPRO_BoardObj
            Pointer to TPRO_LatObj

Returns:    TPRO_INVALID_BOARD_TYPE_ERR - invalid board type for function
            TPRO_COMM_ERR               - error communicating with driver
            TPRO_SUCCESS                - success
```

TPRO_getLongitude

unsigned char TPRO_getLongitude(TPRO_BoardObj *hnd, TPRO_LongObj *Longp);

This routine retrieves the longitude information from the /tSAT device.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the TPRO_LongObj

Returns:    TPRO_INVALID_BOARD_TYPE_ERR - invalid board type for function
            TPRO_COMM_ERR               - error communicating with driver
            TPRO_SUCCESS                - success
```

TPRO_getSatInfo

unsigned char TPRO_getSatInfo(TPRO_BoardObj *hnd, TPRO_SatObj *Satp);

This routine retrieves the number of satellites tracked from the tSAT device.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the TPRO_SatObj

Returns:    TPRO_INVALID_BOARD_TYPE_ERR - invalid board type for function
            TPRO_COMM_ERR               - error communicating with driver
            TPRO_SUCCESS                - success
```

TPRO_getTime

unsigned char TPRO_getTime(TPRO_BoardObj *hnd, TPRO_TimeObj *Timep);

This routine retrieves the current time from the TPRO/tSAT device. The seconds value is received as type double.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the TPRO_TimeObj

Returns:    TPRO_COMM_ERR              - error communicating with driver
            TPRO_SUCCESS               - success
```

TPRO_resetFirmware

unsigned char TPRO_resetFirmware(TPRO_BoardObj *hnd);

This routine resets the firmware programmed on the TPRO/tSAT device. This function is for troubleshooting purposes only and should not be used in the main application.

```
Arguments:  Pointer to the TPRO_BoardObj

Returns:    TPRO_COMM_ERR              - error communicating with driver
            TPRO_SUCCESS               - success
```

TPRO_setHeartbeat

unsigned char TPRO_setHeartbeat(TPRO_BoardObj *hnd, TPRO_HeartObj *Heartp);

This routine controls the hearbeat output. The heartbeat output may be a square wave or pulse at various frequencies.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the TPRO_HeartObj

Returns:    TPRO_FREQ_ERR - invalid frequency value
            TPRO_COMM_ERR - error communicating with driver
            TPRO_SUCCESS  - success
```

TPRO_setMatchTime

unsigned char TPRO_setMatchTime(TPRO_BoardObj *hnd, TPRO_MatchObj *Matchp);

This routine drives the match output line high (start time) or low (stop time) when the desired time is met.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the TPRO_MatchObj

Returns:    TPRO_DAY_PARM_ERR  - invalid days parameter (must be 0-366)
            TPRO_HOUR_PARM_ERR - invalid hours parameter (must be 0 - 23)
            TPRO_MIN_PARM_ERR  - invalid minutes parameter (must be 0 - 59)
            TPRO_SEC_PARM_ERR  - invalid seconds paramter (must be 0 - 69)
            TPRO_COMM_ERR      - error communicating with driver
            TPRO_SUCCESS       - success
```

TPRO_setPropDelayCorr

unsigned char TPRO_setPropDelayCorr (TPRO_BoardObj *hnd, int *us);

This routine sets the propagation delay correction factor.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to correction factor in microseconds

Returns:    TPRO_DELAY_PARM_ERR - invalid propagation delay factor
            TPRO_COMM_ERR       - error communicating with driver
            TPRO_SUCCESS        - success
```

TPRO_setTime

unsigned char TPRO_setTime(TPRO_BoardObj *hnd, TPRO_TimeObj *Timep);

This routine sets the time on the on-board clock of the TPRO/tSAT device. If the board is synchronized to a GPS antenna this value will not be accepted.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the TPRO_TimeObj

Returns:    TPRO_DAY_PARM_ERR  - invalid days parameter (must be 0-366)
            TPRO_HOUR_PARM_ERR - invalid hours parameter (must be 0 - 23)
            TPRO_MIN_PARM_ERR  - invalid minutes parameter (must be 0 - 59)
            TPRO_SEC_PARM_ERR  - invalid seconds paramter (must be 0 - 69)
            TPRO_COMM_ERR      - error communicating with driver
            TPRO_SUCCESS       - success
```

TPRO_setYear

unsigned char TPRO_setYear(TPRO_BoardObj *hnd, unsigned short *yr);

This routine programs the TPRO/tSAT device with the desired year. If the board is synchronized to a GPS antenna this value will not be accepted.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the desired year

Returns:    TPRO_INVALID_BOARD_TYPE_ERR – invalid board type for function
            TPRO_COMM_ERR               - error communicating with driver
            TPRO_SUCCESS                - success
```

TPRO_simEvent

unsigned char TPRO_simEvent(TPRO_BoardObj *hnd);

This routine simulates an external time tag event.

```
Arguments:  Pointer to the TPRO_BoardObj

Returns:    TPRO_COMM_ERR - error communicating with driver
            TPRO_SUCCESS  - success
```

TPRO_synchControl

unsigned char TPRO_synchControl(TPRO_BoardObj *hnd, unsigned char *enbp);

This routine commands the TPRO/tSAT device to synchronize to input or freewheel. This distinction is made using the enable argument. If the enable argument is (0) the clock will freewheel, otherwise it will synchronize to input. When disabling synchronization (freewheeling), the device will continue to synchronize until the time is set.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the synch enable

Returns:    TPRO_COMM_ERR - error communicating with driver
            TPRO_SUCCESS  - success
```

TPRO_synchStatus

unsigned char TPRO_synchStatus(TPRO_BoardObj *hnd, unsigned char *status);

This routine reports the synchronization status of the TPRO/tSAT device. When status is equal to zero, the device is freewheeling. Otherwise the device is synchronized to its input.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the synch status variable

Returns:    TPRO_COMM_ERR - error communicating with driver
            TPRO_SUCCESS  - success
```

TPRO_waitEvent

unsigned char TPRO_waitEvent(TPRO_BoardObj *hnd, TPRO_WaitObj *waitp);

This routine reports the time of an external time-tagged event from the on-board FIFO. Events are stored in the FIFO and this routine will read the FIFO for events. If the FIFO is empty, the routine will wait for an interrupt and report the event for the timeout specified in the TPRO_WaitObj object.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the WaitObj

Returns:    TPRO_COMM_ERR    - error communicating with driver
            TPRO_TIMEOUT_ERR - timeout waiting on event interrupt
            TPRO_SUCCESS     - success
```

TPRO_waitHeartbeat

unsigned char TPRO_waitHeartbeat(TPRO_BoardObj *hnd, int *jiffies);

This routine will report the status of the heartbeat interrupt. Given an amount of time in jiffies, the return code will report timeout or success determined by the heartbeat interrupt status.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the jiffies timeout value

Returns:    TPRO_COMM_ERR    - error communicating with driver
            TPRO_TIMEOUT_ERR - timeout waiting on heartbeat interrupt
            TPRO_SUCCESS     - success
```

TPRO_waitMatch

unsigned char TPRO_waitMatch(TPRO_BoardObj *hnd, int *jiffies);

This routine will report the status of the match time interrupt. Given an amount of time in jiffies, the return code will report timeout or success determined by the match time interrupt status.

```
Arguments:  Pointer to the TPRO_BoardObj
            Pointer to the jiffies timeout value

Returns:    TPRO_COMM_ERR    - error communicating with driver
            TPRO_TIMEOUT_ERR - timeout waiting on match time interrupt
            TPRO_SUCCESS     - success
```

***This page intentionally left blank*.**

Warranty

KSI warrants that all products manufactured by KSI conform to published DSP*Con* specifications and are free from defects in materials and workmanship for a period of one (1) year from the date of delivery when used under normal conditions and within the service conditions for which they were furnished.

The obligation upon KSI arising from a warranty claim shall be limited to repairing, or, at its option, replacing without charge, any product which, in KSI's sole opinion, proves to be defective within the scope of this warranty.

KSI must be notified in writing of the defect or nonconformity within the warranty period, and the affected product must be returned to KSI within thirty (30) days after discovery of such defect or nonconformity.

The buyer shall prepay shipping charges, taxes, duties and insurance for products returned to KSI for warranty service. KSI shall pay for the return of products to buyer except for products returned to another country or from outside the forty-eight contiguous United States.

KSI shall have no responsibility for any defect or damage caused by improper installation, unauthorized modification, misuse, neglect, inadequate maintenance, accident or for any product that has been repaired or altered by anyone other than KSI or its authorized representatives.

The warranty described above is the buyer's sole and exclusive remedy and no other warranty, whether oral or written, is expressed or implied KSI specifically disclaims fitness for a particular purpose. Under no circumstances shall KSI be liable for any direct, indirect, special, incidental or consequential damages, expenses, losses or delays (including loss of profits) based on contract, tort, or any legal theory.

This warranty is valid for a period of one (1) year from the date of shipment. Upon warranty expiration, all services on KSI hardware and software are subject to a current hourly rate, plus travel expenses where applicable. As an option, KSI offers extended warranties when purchased within ten (10) days of receipt of product.

Extended Warranties

HARDWARE

KSI offers an extended warranty on its hardware at a cost of 15% of the current list price per year of coverage. This coverage is available for KSI hardware only.

SOFTWARE

KSI offers an extended warranty on its software at a cost of 15% of the current list price per year of coverage. This coverage is available for KSI software only.

Service and Repair

Before returning a product for service and repair, please contact KSI at (866) KSI-KSI3 to obtain a Return Material Authorization (RMA) number. Have available the Model and Serial numbers of the item, a description of the problem, your company name, and the name and address of the person returning the product.

KSI
a division of DSP*Con*, Inc.

11209 Armour Drive
El Paso, TX 79935

Toll Free Phone: **866-KSI-KSI3**
Toll Free Fax: **866-593-2080**
Website: **www.ksi-corporation.com**

For technical support, contact:

866-KSI-KSI3
pxd@ksi-corporation.com

A Publication of
KSI